Generating Exponentially Smaller POMDP Models Using Conditionally Irrelevant Variable Abstraction

Trey Smith, David R. Thompson, and David S. Wettergreen
Robotics Institute, Carnegie Mellon University, Pittsburgh, PA
{trey,drt,dsw}@ri.cmu.edu

Abstract

The state of a POMDP can often be factored into a tuple of n state variables. The corresponding flat model, with size exponential in n, may be intractably large. We present a novel method called conditionally irrelevant variable abstraction (CIVA) for losslessly compressing the factored model, which is then expanded into an exponentially smaller flat model in a representation compatible with many existing POMDP solvers. We applied CIVA to previously intractable problems from a robotic exploration domain. We were able to abstract, expand, and approximately solve POMDPs that had up to 10^{24} states in the uncompressed flat representation.

Introduction

An agent planning in the real world often faces uncertainty about the state of the world and about the future effects of its actions. Domains with these types of uncertainty can be accurately modeled as partially observable Markov decision processes (POMDPs). The general problem of exactly solving POMDPs is known to be intractable, but recently developed approximation techniques can often find good policies for relatively large POMDPs, depending on how the problem is structured (Spaan & Vlassis, 2005; Pineau & Gordon, 2005).

The state of a POMDP can often be factored into a tuple of n state variables. For example, in a robotic exploration problem, each cell in the map may have a corresponding variable whose value represents the contents of the cell. The corresponding unfactored or "flat" model has state space size exponential in n, which may be intractably large. This issue is important because most existing POMDP solvers operate on a flat model representation, with only a few exceptions (Hansen & Feng, 2000; Poupart & Boutilier, 2004).

However, in some problems there are efficient ways to identify irrelevant variables that cannot affect the solution. In that case the irrelevant variables can be abstracted away, exponentially shrinking the state space in the flat model (Boutilier & Dearden, 1994). If the overall task can be hierarchically decomposed into subtasks, one can take a finer-grained approach and temporarily abstract away variables that are relevant overall but irrelevant within a particular subtask (Pineau, Gordon, & Thrun, 2003). When interleaving planning and execution, the amount of abstraction may also vary at different planning horizons (Baum & Nicholson, 1998).

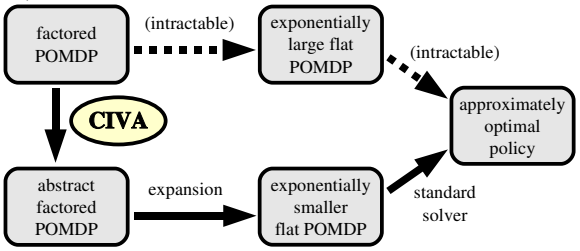

Figure 1: CIVA Process Diagram.

We present an alternative method called conditionally irrelevant variable abstraction (CIVA) for losslessly reducing the size of the factored model. A state variable is said to be *conditionally irrelevant* for a given partial assignment to other state variables if certain conditions are satisfied that guarantee it can be temporarily abstracted away without affecting policy optimality. Figure 1 shows how CIVA fits into the overall planning process. Our method considers only factored state, although factored actions and observations can also be useful (Guestrin, Koller, & Parr, 2001; Feng & Hansen, 2001).

We applied CIVA to previously intractable POMDPs from a robotic exploration domain. We were able to abstract, expand, and approximately solve POMDPs that had up to 10^{24} states in the uncompressed flat representation. The resulting policies outperformed hand-tuned heuristic policies both in simulation and in testing onboard a robot in a controlled outdoor environment.

Example Problem

Our primary testing domain for CIVA was the *LifeSurvey* robotic exploration problem. We will use *MiniLifeSurvey*, a simplified version of *LifeSurvey*, to provide intuition about conditional irrelevance. In *MiniLifeSurvey*, a robot is moving through a one-dimensional map from west to east. The robot has sensors for detecting life en route, but it must balance the cost of using these sensors against the expected value of the resulting data. The robot has three available actions:

1. `move`: Moves the robot one cell to the east, with a cost of -1. Always returns a `null` observation.

2. `scan`: Applies the robot's long-range sensor, providing noisy information as to whether life is present in the cell just ahead of the robot, with a cost of -2. Returns either a `positive` or `negative` observation.
3. `sample`: Applies the robot's short-range sensor to the current cell, with a cost of -10. Returns either a `positive` or `negative` observation. If the cell contains detectable life, the robot receives a reward of +20.

The variables in *MiniLifeSurvey* are:

1. X_1: The position of the robot, ranging from 1 to k. The position is always known and advances deterministically when the `move` action is applied.
2. $Y_1, \ldots, Y_k$: Each Y_i has the value `L` or `N` ("life" or "no life") depending on whether cell i of the map contains detectable life or not.

The robot starts in cell 1 and has remote sensing data that provides independent prior probabilities for the Y_i variables. The problem ends when the robot applies the `move` action in the rightmost cell.

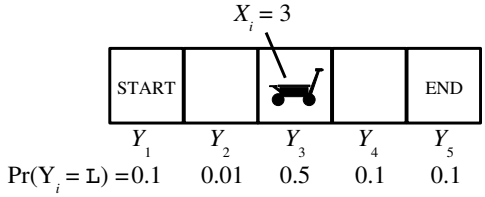

Figure 2: The *MiniLifeSurvey* problem.

Figure 2 shows an instance of *MiniLifeSurvey* with $k = 5$. The priors $\Pr(Y_i = \texttt{L})$ for each Y_i are shown below the map. The robot is shown at position $X_1 = 3$.

The key insight underlying CIVA is that in a structured problem like *MiniLifeSurvey* the robot only needs to consider joint assignments to a few of the state variables at any one time. In position $X_1 = 3$, only the variables Y_3 and Y_4 are immediately relevant in the sense that they can affect the rewards or observations in the next time step. Because the robot only moves forward, variables Y_1 and Y_2 can have no further effect on the system. Variable Y_5 will be important later, but its value cannot be influenced by any of the other Y_i variables or the robot's action, so in considering the next action to take the robot can temporarily disregard Y_5 and reconstruct its probability distribution later as needed. (These concepts will be formalized later.)

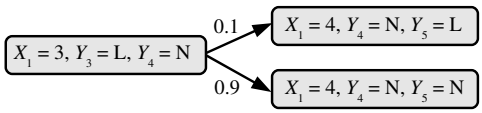

Figure 3: A state transition in the abstract model.

Figure 3 shows one example state transition for the `move` action in the abstract model produced by CIVA. Each abstract state in the reduced model corresponds to an equivalence class of states in the original model. For example, the abstract state on the left corresponds to the set of all states with $X_1 = 3$, $Y_3 = \texttt{L}$, $Y_4 = \texttt{N}$, and any value for the other Y_i variables. The arrows in the diagram are labeled with transition probabilities.

As we will explain later, the abstract states in the abstract model specify values only for the Y_i variables that are conditionally relevant given the value of X_1. With $X_1 = 3$, Y_3 and Y_4 are conditionally relevant; with $X_1 = 4$, Y_4 and Y_5 are conditionally relevant.

Abstracting away variables in the factored model results in an exponentially smaller flat model. In the uncompressed *MiniLifeSurvey* model with map length k, there are k possible values for X_1 and k binary-valued variables Y_i, so there are $k \times 2^k$ states. In the CIVA-compressed model, only the position and two of the Y_i variables need to be tracked at a time, so there are just $4k$ abstract states.

POMDP Background

A POMDP is described by a tuple $P = \langle \mathcal{S}, \mathcal{A}, T, R, \gamma, \mathcal{O}, O, b_0 \rangle$. $\mathcal{S}$ is a finite set of world states. $\mathcal{A}$ is a finite set of actions available to the agent. T is a transition function such that $T(s, a, s')$ is the probability of transitioning from state s to state s' when applying action a. R is a reward function such that $R(s, a)$ is the immediate reward received by the agent for taking action a in state s. $\gamma < 1$ is the discount factor. $\mathcal{O}$ is a finite set of observations. O is an observation function such that $O(a, s', o)$ is the probability of receiving observation o if the agent applies action a and the world transitions to state s'. b_0 is the agent's initial belief, such that $b_0(s)$ is the probability that the initial state is s.

A POMDP policy is a mapping from histories to actions in the form

$$a_t = \pi(a_0, o_0, a_1, o_1, \ldots, a_{t-1}, o_{t-1}). \tag{1}$$

Given a system model and the initial belief b_0, the agent can use Bayesian updates to calculate the posterior belief b_t corresponding to any history and rewrite the policy in the form $a_t = \pi(b_t)$. It is a theorem that every POMDP has an optimal policy π^*, which among all policies π maximizes the long-term expected reward

$$J^P \pi(b) = E_{\pi, b_0 = b} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right] \tag{2}$$

for all beliefs b. Solving the POMDP exactly means finding such an optimal policy, but most practical algorithms find a policy that is only guaranteed to be near-optimal for a given starting belief b_0.

We say that two models $P = \langle \mathcal{S}, \mathcal{A}, T, R, \gamma, \mathcal{O}, O, b_0 \rangle$ and $P' = \langle \mathcal{S}', \mathcal{A}', T', R', \gamma', \mathcal{O}', O', b_0' \rangle$ are *policy-compatible* if $\mathcal{A} = \mathcal{A}'$ and $\mathcal{O} = \mathcal{O}'$. Policy compatibility ensures that any policy for P' can be applied to P, since policies for the two models have the same functional form according to equation (1), although they may represent system state and beliefs differently.

We say that P and P' are *policy-equivalent* if for every policy π the long-term expected reward of the initial belief is the same for the two models

$$J^P \pi(b_0) = J^{P'} \pi(b_0'). \tag{3}$$

Policy equivalence ensures that the two models have the same optimal policies.[1] Applying CIVA to a model P produces a policy-equivalent model P'.

Conditional Relevance

CIVA is based on the assumption that the state can be factored into a tuple of discrete variables $\langle X_1, \ldots, X_j, Y_1, \ldots, Y_k \rangle$, where $X_1, \ldots, X_j$ are *upstream variables* and $Y_1, \ldots, Y_k$ are *downstream variables*. We denote the tuple of upstream values $X = \langle X_1, \ldots, X_j \rangle$, and similarly $Y = \langle Y_1, \ldots, Y_k \rangle$. We use ϕ to denote the value of particular variables; for example, $\phi_{X_i}(s)$ is the value of variable X_i in state s, and $\phi_X(s)$ is the joint value of all upstream variables in state s.

Upstream variable values are always known to the agent, transition deterministically, and are independent of the downstream variables. Their dynamics are already specified along with the other variables as part of the transition function T, but it is also convenient to define special notation that reflects the additional structure. We define the upstream transition function U such that $x' = U(x, a)$, where x is the value of the upstream variables at one time step, and x' is the upstream value at the next time step after taking action a.

An upstream value x is *reachable* if starting from the known initial upstream value x_0 there is a sequence of upstream transitions that reaches x. The set of all reachable upstream values can easily be generated by forward recursion from x_0.

We will build up the definition of conditional relevance in several steps. A downstream variable Y_i is *immediately relevant* at x, written $Y_i \in f^{\text{IR}}(x)$, if it is possible for Y_i to have an "immediate effect" on the problem. That is, $Y_i \in f^{\text{IR}}(x)$ unless all of the following constraints hold:

I1. Y_i has no immediate effect on reward. For any state s, let s/E denote the set of states that agree with s over all variables other than Y_i. Let a be an action, let s be a state with $\phi_X(s) = x$, and let $s' \in s/E$. Then we must have $R(s, a) = R(s', a)$.

I2. Y_i has no immediate effect on observations. Let a be an action, o be an observation, let s be a state with $\phi_X(s) = x$, and let $s' \in s/E$. Then we must have $O(a, s, o) = O(a, s', o)$.

I3. Y_i has no immediate effect on the transitioning of other variables. Let a be an action, let s be a state with $\phi_X(s) = x$, let $s' \in s/E$, and let s'' be an arbitrary state. Then we must have

$$\sum_{\sigma \in s''/E} T(s, a, \sigma) = \sum_{\sigma \in s''/E} T(s', a, \sigma). \qquad (4)$$

The idea is that immediately relevant variables tend to become "entangled" with the rest of the system, so they typically cannot be abstracted away without losing policy-equivalence.

[1] These definitions of policy compatibility and equivalence are novel as far as we know.

A downstream variable Y_i is *a-predictable* at x, written $Y_i \in f^{\text{P}}(x, a)$, if it is possible to reconstruct the distribution over possible values of Y_i after applying action a in a state with $\phi_X(s) = x$, given only knowledge of x, a, and b_0. In other words, given x, a, and b_0, the distribution over possible values for Y_i after the transition must be conditionally independent of all other state information, including the preceding value of Y_i and other downstream variables. The idea is that we can temporarily "forget" probabilistic information about the value of a variable, even one that is going to be relevant later, if at that later point the information can be reconstructed.

A downstream variable Y_i is *conditionally relevant* at x, written $Y_i \in f^{\text{R}}(x)$, if either:

C1. The variable Y_i is immediately relevant at x, or

C2. For some action a, (i) Y_i is conditionally relevant at $x' = U(x, a)$, and (ii) Y_i is not a-predictable at x.

The idea is that the agent needs to keep track of probabilistic information about a variable if it is either immediately relevant or there is a future point where it is both relevant and we have no way to reconstruct the information.

Relevance Determination

We assume that the factored model provided to CIVA specifies which variables are upstream versus downstream, as this is easy to determine manually. However, we still need a way to determine the upstream values where downstream variables are conditionally irrelevant so we can abstract them away. We call this problem *relevance determination*.

A relevance determination algorithm is *exact* if for any upstream value x it calculates the exact set $f^{\text{R}}(x)$ as defined above. In contrast, it is *conservative* if for any x it calculates a superset of $f^{\text{R}}(x)$. In other words, a conservative algorithm errs only on the side of tagging variables relevant when they are not. Conservative relevance determination may result in a compressed model that is larger than necessary, but it retains the key property that the original model and compressed model are policy-equivalent.

Our approach to relevance determination is conservative. It is a three-step process. We (1) find immediately relevant variables, (2) find predictable variables, and (3) find conditionally relevant variables.

Finding Immediately Relevant Variables

The first step in relevance determination is to calculate the immediately relevant variables $f^{\text{IR}}(x)$ for every reachable upstream value x. To ensure that the overall conditional relevance determination is conservative, the immediate relevance determination also needs to be conservative. That is, when in doubt it must tag a variable as immediately relevant.

Checking the immediate relevance constraints I1-I3 for $Y_i \in f^{\text{IR}}(x)$ by brute force is often intractable, since each constraint involves enumeration over the set of all states s with $\phi_X(s) = x$; this set has size exponential in the number of downstream variables. Thus tractable immediate relevance determination depends on leveraging the structure of the factored model.

CIVA is a general approach that is not tied to any particular factored representation. Possible representations for the R, O, and T functions include decision trees (Boutilier, Dearden, & Goldszmidt, 2000) and algebraic decision diagrams (ADDs) (St. Aubin, Hoey, & Boutilier, 2000), among others. Immediate relevance determination can be performed over any representation, but the choice of representation affects its computational complexity.

For concreteness, we describe the process in more detail with a particularly simple decision tree representation. Let the functions R, O, and T be represented as decision trees with the following variable ordering: (1) first branch on the action, (2) then on the observation (for O only), (3) then on upstream state variables in an arbitrary fixed order, (4) then on downstream state variables in an arbitrary fixed order. The T function takes two state arguments s and s'; all state variables of s are ordered before all state variables of s' in the decision tree. Let n_R, n_O, and n_T be the number of nodes in the decision tree representations of R, O, and T respectively.

One can determine if $Y_i \in f^{\text{IR}}(x)$ using the following procedure:

1. Check I1. Restricting the function R to a particular action a and upstream value x corresponds to selecting a particular subtree of the decision tree. If for every action a the corresponding subtree does not contain a node branching on Y_i, then I1 is satisfied. Overall, this check can be performed in order n_R time (and usually much faster, since portions of the tree relating to other upstream values other than x can be ignored).
2. Check I2. Similar to the check of I1, this time iterating over all combinations of actions and observations in O. This check can be performed in order n_O time.
3. Check I3. Restricting T to a particular action a and upstream value $\phi_X(s) = x$ corresponds to selecting a particular subtree. Then in order to check I3 we must (1) sum out the Y_i variable of s' within the subtree, (2) recursively canonicalize the subtree by eliminating branch nodes that have identical subtrees on either branch, and (3) check if the subtree now contains a node branching on the Y_i variable of s. This procedure can be performed for all actions in order n_T time.

Thus, for each upstream value x and variable Y_i, we can check if $Y_i \in f^{\text{IR}}(x)$ in order $n_R + n_O + n_T$ time, which is relatively efficient if the model is compact.

Note that not all problems with factored structure can be compactly represented with this type of decision tree. We expect that efficient conservative immediate relevance determination algorithms exist for ADDs and under certain relaxations of the variable ordering constraints, but this is the only case we have worked out in detail.

Finding Predictable Variables

The second step in relevance determination is to calculate the a-predictable variables $f^{\text{P}}(x, a)$ for every reachable upstream value x and action a. Recall that $Y_i \in f^{\text{P}}(x, a)$ if, after applying action a in a state with $X = x$, it is possible to reconstruct the probability distribution of Y_i given only knowledge x, a, and b_0. Because of the way predictability relates to conditional relevance, we say that a predictability determination algorithm is conservative if it errs only on the side of tagging variables *not* predictable.

We do not know of any tractable algorithm for exact predictability determination in the general case. Predictability of Y_i at x depends on whether or not the agent is able to gain information about Y_i on the way from the initial state to a state with $X = x$, and whether that information is still pertinent when it arrives. Since these considerations can in general depend on the path that the agent takes through the state space, it might be very difficult to check that a variable is predictable over all paths.

However, if the goal is *conservative* predictability determination, there are several types of structure that make it easy to prove that a variable is predictable. For example:

- If applying action a in a state with $X = x$ overwrites all previous information about Y_i, then $Y_i \in f^{\text{P}}(x, a)$. For example, if action a flips a coin, the agent knows that there is a 50% chance the coin shows heads after the state transition, regardless of what information it might have had before.
- If one can show that any path leading to a state with $X = x$ must pass through a state with $X = x'$, and entering a state with $X = x'$ fixes a known and permanent value for Y_i, then $Y_i \in f^{\text{P}}(x, a)$. For example, suppose the only way to get hold of a fire extinguisher is to break its glass case. Then if the agent has the fire extinguisher, it can reconstruct the fact that the glass is broken.

In our robotic exploration domain, we rely on yet another type of special structure that implies predictability. A variable Y_i is *untouched* at x, written $Y_i \in f^{\text{U}}(x)$ if it satisfies:

U1. Y_i is independent of other variables in the initial belief,
U2. The value of Y_i does not change over time,
U3. Y_i is not immediately relevant at x, and
U4. For every predecessor upstream value x' such that $x = U(x', a)$, Y_i is untouched at x'.

If a variable is untouched at x, we can be sure that its probability distribution is unchanged from what it was in b_0. This makes it a-predictable at x for every action a. For example, if the agent starts out believing there is a utility closet upstairs with 50% probability, and the agent has not yet had enough time to go upstairs and check, its current belief about the utility closet is just its initial belief.

We can identify untouched variables using forward recursion. First we mark as touched every variable that violates U1-U3. Then we perform local updates to enforce the consistency of U4; if Y_i is touched at x, then Y_i is marked as touched at all successors $x' = U(x, a)$ of x. Local updates are propagated forward until U4 is globally satisfied.

Finding Conditionally Relevant Variables

The final step in relevance determination is to calculate the conditionally relevant variables $f^{\text{R}}(x)$ for every reachable upstream value x. The reader may wish to review the definition of conditional relevance, conditions C1 and C2 above.

With $f^{\text{IR}}(x)$ and $f^{\text{P}}(x)$ in hand, it is straightforward to calculate $f^{\text{R}}(x)$ by backward recursion. First we mark ev-

ery immediately relevant variable as conditionally relevant to satisfy C1. Then we perform local updates to enforce the consistency of C2. If Y_i is conditionally relevant at x, then it is marked as conditionally relevant for all predecessors x' such that $x = U(x', a)$ and $Y_i \notin f^{\mathrm{P}}(x', a)$. Local updates are propagated backwards until C2 is globally satisfied.

Model Abstraction

This section defines the form of the abstract model produced by CIVA. First we define the *predictability transformed* version of the transition function T. Let x be an upstream value and a be an action such that $Y_i \in f^{\mathrm{P}}(x, a)$, and let s be a state consistent with x. The predictability of Y_i means that its probability distribution after the state transition can be calculated given only knowledge of x, a, and b_0. There are two ways this can happen:

1. For prior states with $X = x$, the value of Y_i after applying a depends only on a. In this case no change needs to be made.
2. The value of Y_i after applying a formally depends on some downstream variable Y_m, but in fact all reachable beliefs with $X = x$ have probability distributions for Y_m that lead to the same prediction of Y_i.[2] In this case, we can rewrite the transition function so that, independent of the value of Y_m, the posterior probability distribution of Y_i is its reconstructed value as an a-predictable variable.

The result of performing this rewrite wherever possible is denoted $\tilde{T}$.

Conditional relevance defines an equivalence relation E on states, such that for two states s, s', we have $E(s, s')$ if s and s' both (i) share the same upstream value x and (ii) agree on the values of the conditionally relevant downstream variables $f^{\mathrm{R}}(x)$. E induces a partition of $\mathcal{S}$ into equivalence classes of similar states. Let $s/E = \{s' \mid E(s, s')\}$ denote the class containing state s.

We will abuse notation by writing versions of $\tilde{T}$, O, and R that take equivalence classes as arguments. We define

$$\tilde{T}(s, a, s'/E) = \sum_{\sigma \in s'/E} T(s, a, \sigma), \tag{5}$$

and we define $\tilde{T}(s/E, a, s'/E) = q$ if for all $\sigma \in s/E$, we have $\tilde{T}(\sigma, a, s'/E) = q$. Otherwise $T(s/E, a, s'/E)$ is not well defined. $R(s/E, a)$ and $O(a, s'/E, o)$ can be well-defined or not in a similar way.

It turns out that with conditional relevance defined as presented earlier, for all s, s', a, o, we have that $\tilde{T}(s/E, a, s'/E)$, $R(s/E, a)$ and $O(a, s'/E, o)$ are well-defined. Thus equivalence classes in the original model can be used as states in the reduced model, and the equivalence-class versions of $\tilde{T}$, R, and O define the reduced system dynamics. The fact that the reduced system dynamics are well-defined implies that the abstract model is policy-equivalent to the original. We include only the equivalence classes corresponding to reachable upstream values in the reduced model.

[2] When we say a belief with $X = x$ we mean a belief in which only states with $X = x$ have non-zero probabilities.

Application to *MiniLifeSurvey*

Now we tie some of the formal concepts back to the *MiniLifeSurvey* domain introduced earlier.

When the robot is at position $X_1 = 3$, the immediately relevant variables are $f^{\mathrm{IR}}(x) = \{Y_3, Y_4\}$. Cell 3 is the current cell, so Y_3 affects the observation and the reward when applying the `sample` action. Cell 4 is the cell just ahead of the robot, so Y_4 affects the observation when applying the `scan` action.

Recall that in general all untouched variables are a-predictable. In *MiniLifeSurvey*, the converse happens to be true as well. With $X_1 = 3$, the only untouched variable is $f^{\mathrm{U}}(x) = \{Y_5\}$. As the robot moves from west to east, all other downstream variables have already had a chance to affect observations.

With $X_1 = 3$, the conditionally relevant variables are $f^{\mathrm{R}}(x) = \{Y_3, Y_4\}$. Y_1 and Y_2 are irrelevant because there is no way they can affect future observations or rewards. Y_5 is irrelevant because, even though it will become immediately relevant when $X_1 = 4$, it is currently untouched.

Figure 3 shows one example state transition for the `move` action in the abstract model. The value of Y_4 naturally remains the same across the transition, since all the Y_i variables are static. This could be inferred directly from the original transition function T. On the other hand, the value of Y_5 was not specified before the transition. The distribution over Y_5 values after the transition is inferred from the prior probability information $\Pr(Y_5 = \mathtt{L}) = 0.1$ from the initial belief. This information from b_0 was effectively folded into the transition function when T was transformed into $\tilde{T}$.

Related Work

Boutilier & Dearden (1994) present a notion of globally irrelevant variables for MDPs; CIVA's conditional irrelevance is finer-grained, offering more opportunities for abstraction. Baum & Nicholson (1998) suggest a non-uniform abstraction like that of CIVA, but they assume a context of interleaving planning and execution, and they vary the level of detail at different planning horizons.

Some approaches directly operate on the factored model during the POMDP solution process. McAllester & Singh (1999) represent beliefs compactly using lossy Boyen-Koller belief simplification. St. Aubin, Hoey, & Boutilier (2000) developed an ADD representation for MDPs, extended by Hansen & Feng (2000) to apply to POMDPs and use a factored observation model.

When considering factored solvers like these, one must balance the benefits of a compact representation against the additional code complexity and overhead of operations on factored data structures. For problems like *LifeSurvey*, CIVA can provide enough abstraction that operating on the flat model becomes tractable. This makes the problem compatible with efficient flat POMDP solvers, avoiding the need for a factored solver.

In other cases, CIVA abstraction might be useful as a preprocessor for a factored solver. Some of the abstraction performed explicitly by CIVA is already captured implicitly with an efficient factored representation; it is currently an open question whether CIVA abstraction can significantly speed up operations in the resulting factored model.

Givan, Dean, & Greig (2003) present a unifying theoretical framework and useful notation that encompasses many approaches to state aggregation in MDPs and POMDPs. CIVA could be accommodated by a slight extension of their framework, taking into account initial belief information.

Pineau, Gordon, & Thrun (2003) provide just one example of a number of approaches that use subtasks or macros to facilitate abstraction. CIVA abstraction relies on other kinds of structure, such as locality of variable effects and forward progress rendering some variables irrelevant.

Poupart & Boutilier (2004) present a linear compression technique that can losslessly capture some of the same structure as CIVA, and they also go further in providing good lossy compressions. However, their approach apparently does not leverage initial belief information, and in any case the method is so different from CIVA that insight can be gained by studying both.

Robotic Exploration Task

The *LifeSurvey* problem is motivated by advances in planetary surface robotics. Future robots will be able to move several kilometers through unexplored terrain in a single command cycle. They will explore much more efficiently if they have a number of capabilities that fall under the general rubric of "science autonomy" (Castaño *et al.*, 2003).

An exploring robot should use any available orbital data to focus its scanning effort and ensure its path leads through the most scientifically promising regions. If it detects a target of opportunity, it should automatically move closer and take more detailed follow-up measurements. All of these behaviors are included in the *LifeSurvey* problem, and the POMDP framework allows us to model both initial uncertainty and noisy observations collected during execution.

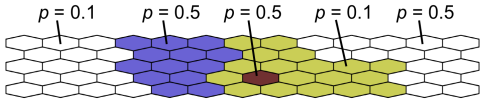

Figure 4: A *LifeSurvey* prior map.

The full *LifeSurvey* problem places the robot in a two-dimensional map. The robot must move from the west edge of the map to the east edge, but within the bounds of the map it could choose its own path. Figure 4 shows an example prior map. Differently shaded regions of the map have different per-cell prior probabilities of containing detectable life. In addition, the robot receives reward only the first time it samples a cell with evidence of life in any given region.

We tested the performance of *LifeSurvey* policies both in simulation and onboard a robot. Our robotic platform was Zoë, a capable exploration robot developed as part of the Life in the Atacama project, a three year effort to test high-mobility science strategies for robotic astrobiology in the Atacama Desert of Chile (Dohm *et al.*, 2005). Our *LifeSurvey* testing with Zoë was conducted in a controlled outdoor environment, shown in Figure 5. Small artificial markers were used as stand-ins for signs of life, and scanning and sampling actions were implemented using the robot's onboard cameras.

Figure 5: The Zoë rover executing a *LifeSurvey* policy.

In a single *LifeSurvey* action, the rover could either (1) scan the three forward cells to the northeast, east, and southeast, returning a noisy signal as to whether they contain life, or (2) perform a simple move or sampling move to any one of the forward cells. Sampling moves differed from simple moves in that they caused the rover to take additional detailed measurements as it entered the new cell. They were intended to confirm the presence of life.

The observation returned by the scan action was a tuple of three independent readings, one for each forward cell. The possible values for each reading could be interpreted roughly as "negative", "maybe", or "positive". (CIVA did not make use of the factored structure of the observations; as far as it was concerned, each scan action simply returned one of $3^3 = 27$ possible flat observations.)

The different possible values corresponded to different confidence levels from the onboard detection routine searching for artificial markers. The sensor noise parameters used in the planning model were learned from a training set that included detection routine outputs and ground truth labels gathered over several runs in the testing environment. Cells without markers returned negative/maybe/positive readings roughly 72%/12%/16% of the time, respectively; cells with markers had the distribution 9%/5%/86%.

The planning objective was to maximize expected reward. The robot received a per-region reward: +5 points if the robot entered the region, +20 points if it passed through a life-containing cell in the region, or +50 points if it performed a sampling move into a life-containing cell in the region. Each action incurred a cost: -1 point for each move, and -5 points for each scan or sampling move. Thus the rover needed to find confirmed evidence of life in as many regions as possible, while minimizing the number of detours, scans, and sampling moves.

In the interest of expediency we developed a special-purpose version of CIVA for *LifeSurvey*. The uncompressed system dynamics were expressed procedurally, rather than in a declarative representation like a decision tree or ADD. As the immediate relevance structure for *LifeSurvey* was fairly simple, we found it easiest to provide CIVA with a hard-coded conservative labeling of immediately relevant vari-

ables rather than writing a general-purpose routine to check I1-I3. Determination of predictability and conditional relevance used constraint propagation as described earlier.

The *LifeSurvey* problem is well suited to CIVA. The downstream variables in *LifeSurvey* include both per-cell variables (presence or absence of life), and per-region variables (has life been sampled in this region yet?). Only the robot's current cell and cells just ahead are relevant, and regions that the robot has permanently left behind or has yet to encounter are irrelevant.

The instance of *LifeSurvey* shown in Figure 4 had 3.5×10^{24} states in the unreduced flat representation versus 7,001 with the flat representation after CIVA. The majority of the compression, a factor of about 5.8×10^{17}, came from abstracting away irrelevant per-cell variables. This factor would be roughly the same for any map with the same number of cells. Abstracting away irrelevant per-region variables compressed by another factor of about 200. This factor depends on the shape of the regions; in the worst case, if regions were arranged such that the robot could drive from any region to any other, all per-region variables might be relevant simultaneously, which would lead to very little compression. The two sources of abstraction together reduced the size of the model to 29,953 states, of which 7,001 were reachable.

All of our computation was performed on a 3.2 GHz Pentium-4 processor with 2 GB of main memory. Less than two seconds were required to generate and write out the compressed flat model.

We approximately solved the compressed *LifeSurvey* problem using the freely available ZMDP solver for POMDPs (Smith, 2006). Specifically, ZMDP used the FRTDP heuristic search algorithm in conjunction with generalizing representations for value-function bounds developed for the HSVI2 algorithm (Smith & Simmons, 2005). After 10^3 seconds of wallclock time, the solver was able to produce a policy whose expected long-term reward was guaranteed to be within 20% of the optimal policy. In contrast, the uncompressed flat model would have been several orders of magnitude too large to fit in memory.

Experimental Evaluation

We evaluated three policies on the *LifeSurvey* problem. Under the blind policy, the rover simply moved to the right in a straight line, always using sampling moves. The blind policy would confirm the presence of life only if it was found on the straight-line path.

Under the reactive policy, the rover followed a set of simple hand-generated rules designed to efficiently confirm the presence of life through combined use of scanning and sampling. It moved forward through the map, performing a scan action after each move, and detoured to take a sampling move if life was detected in a scan. When life was not detected, the reactive policy tried to stay on a preplanned path that was optimized to pass by areas likely to contain life. This is the kind of policy that a domain expert might generate without use of AI planning techniques.

The third policy was the approximately optimal policy output by POMDP planning using the CIVA-compressed model.

The reactive and probabilistic policies were each evaluated on 20 runs through the test course; there were 2 prior maps, times 2 randomly drawn target layouts per map, times 5 runs per target layout. The blind policy could be evaluated on the same target layouts in simulation since its actions did not depend on uncertain sensor observations.

Policy	Search acts	Regions confirmed	Reward
Blind	12.0	2.5	68
Reactive	20.0	**3.4**	61
POMDP	**7.5**	3.0	**113**

Figure 6: *LifeSurvey* experimental performance.

Results are shown in Figure 6. For each policy, we report average values over the 20 runs. "Search acts" gives the number of scan and sampling move actions used per run (smaller values are better). "Regions confirmed" gives the number of regions in which the presence of life was confirmed with a sampling move action (higher values are better). Finally, "Reward" is the combined efficiency metric that we were trying to optimize (higher values are better).

The POMDP policy performed best in terms of search actions and mean reward, by statistically significant margins. The reactive policy confirmed the presence of life in more regions, but at the cost of far more search time than the other policies. The same ordering was also observed in simulation results with a much larger number of trials.

The purpose of this experiment was not to demonstrate that the particular POMDP policy we tested was ideal. We could certainly have generated a better policy by increasing the amount of time allotted to POMDP planning. With sufficient effort coding and testing, we could probably even find a manually-tuned heuristic policy that would outperform the approximately optimal POMDP policy. Rather, the purpose was to show that CIVA compression made it possible to formulate *LifeSurvey* using the highly expressive POMDP modeling framework, and within that framework to generate a policy with a strong quality guarantee and competitive performance.

Conclusions

We presented CIVA, a novel approach for losslessly compressing POMDPs with appropriate factored structure. When applied to the *LifeSurvey* robotic exploration domain, we were able to abstract and approximately solve POMDPs whose uncompressed flat representation had up to 10^{24} states.

Effective use of CIVA relies on strong assumptions about problem structure. There must be deterministic state variables to use as upstream variables. If the *LifeSurvey* model included position uncertainty, position could not have been used as an upstream variable.

The existence of conditionally irrelevant variables tends to rely on a sense of "forward progress" through the system. If the *LifeSurvey* robot was able to move backward through the map, cells it passed by would no longer be irrelevant. If the problem was cyclical in nature, there would typically be no untouched state variables after the first cycle. (Although

some variables might still be a-predictable due to other types of structure.)

Irrelevance also requires a certain amount of independence between downstream variables. If downstream variables are correlated in the b_0 prior, then they cannot be considered untouched according to our current definition (although the requirements could be relaxed in some circumstances). Similarly, downstream variables can become entangled if they affect each other's transitions or they jointly affect observations.

For these reasons, we expect that only a small proportion of interesting POMDP problems would gain significant benefit from the full CIVA compression algorithm described here. However, many more problems have approximate conditional irrelevance structure which could lend itself to lossy compression extensions of CIVA.

Overall, the reader may wish to think of this paper less as a description of an integrated algorithm and more as a conceptual toolkit. Depending on the problem, some or all of the CIVA concepts may be applicable. For instance, the idea of rewriting the transition function based on some form of reachability analysis in order to fold in information from the initial belief and remove dependencies on the prior state may work with other types of problem structure that we have not considered.

References

Baum, J., and Nicholson, A. E. 1998. Dynamic non-uniform abstractions for approximate planning in large structured domains. In *Pacific Rim Int. Conf. on Artificial Intelligence*.

Boutilier, C., and Dearden, R. 1994. Using abstractions for decision-theoretic planning with time constraints. In *AAAI*.

Boutilier, C.; Dearden, R.; and Goldszmidt, M. 2000. Stochastic dynamic programming with factored representations. *Artificial Intelligence* 121:49–107.

Castaño, R.; Anderson, R. C.; Estlin, T.; DeCoste, D.; Fisher, F.; Gaines, D.; Mazzoni, D.; and Judd, M. 2003. Rover traverse science for increased mission science return. In *IEEE Aerospace Conf.*

Dohm, J.; Cabrol, N.; Grin, E.; Moersch, J.; Diaz, G.; Cockell, C.; Coppin, P.; Fisher, G.; Hock, A.; Marinangeli, L.; Minkley, E.; Ori, G.; Piatek, J.; Warren-Rhodes, K.; Weinstein, S.; Wyatt, M.; Smith, T.; Wagner, M.; Stubbs, K.; Thomas, G.; and Glasgow, J. 2005. Life in the Atacama year 2: Geologic reconnaissance through long-range roving and implications on the search for life. In *Lunar and Planetary Sci. Conf.*

Feng, Z., and Hansen, E. 2001. Approximate planning for factored POMDPs. In *European Conf. on Planning*.

Givan, R.; Dean, T.; and Greig, M. 2003. Equivalence notions and model minimization in Markov decision processes. *Artificial Intelligence* 147:163–223.

Guestrin, C.; Koller, D.; and Parr, R. 2001. Multiagent planning with factored MDPs. In *NIPS*.

Hansen, E., and Feng, Z. 2000. Dynamic programming for POMDPs using a factored state representation. In *ICAPS*.

McAllester, D., and Singh, S. 1999. Approximate planning for factored pomdps using belief state simplification. In *UAI*.

Pineau, J., and Gordon, G. 2005. POMDP planning for robust robot control. In *Int. Symp. on Robotics Res. (ISRR)*.

Pineau, J.; Gordon, G.; and Thrun, S. 2003. Policy-contingent abstraction for robust robot control. In *UAI*.

Poupart, P., and Boutilier, C. 2004. VDCBPI: an approximate scalable algorithm for large scale POMDPs. In *NIPS*.

Smith, T., and Simmons, R. 2005. Point-based POMDP algorithms: Improved analysis and implementation. In *UAI*.

Smith, T. 2006. ZMDP software for POMDP and MDP planning. http://www.cs.cmu.edu/~trey/zmdp/.

Spaan, M. T. J., and Vlassis, N. 2005. Perseus: Randomized point-based value iteration for POMDPs. *J. Artificial Intelligence Res.* 24:195–220.

St. Aubin, R.; Hoey, J.; and Boutilier, C. 2000. APRICODD: Approximate policy construction using decision diagrams. In *NIPS*, 1089–1095.